Processors and Architectures for Embedded Systems

General Purpose vs. Application Specific Processors

Lectures 5 & 6 Overview

• General Purpose vs. Application Specific Processors
• Core (IP)-based design
• Reconfigurable Systems

Agenda

• Trends in Embedded-Microprocessor design
• Embedded System Architectures
• Customized Instruction Sets for Embedded Processors
• Selected Co-design problems
• Conclusions

Embedded Microprocessors Overview

• Desktop vs. Embedded processors
• Microcontrollers and Microprocessors
• New applications drive requirements
  – game consoles
  – handheld, palmtop, automobile and network PCs
  – cellular phones and other mobile communicators
  – modems, fax machines, printers, etc.
  – digital cameras

Comparing Embedded Processors

• Power consumption
• Code density
• Peripheral integration and chipsets
• Multimedia acceleration
• Price/performance ratio

Standardization

• The heterogeneity of embedded architectures
• A need to unify the embedded processor market?
• Windows CE and JAVA as examples
• Will management influence standardization?
Introduction to ES Architectures

- ES Architectures are determined by:
  - circuit technology;
  - application requirements;
  - market constraints:
    - strict cost margins
    - time-to-market and predictable design time
    - (hard) time constraints
    - power dissipation
    - safety
    - physical constraints

Components and Systems

- Components as function blocks
- Systems as large entities of integrated components
- Single control thread vs. multiple control threads
- Architecture specialization
  - component specialization techniques
  - system specialization techniques

Components and Systems

- Load distribution
  - Control decomposition (control clustering)
  - Data decomposition (data clustering)
- Component interaction

Component Specialization Techniques

- Instruction set specialization
- Function unit and data path specialization
- Memory specialization
- Interconnect specialization
- Control specialization

System Specialization Techniques

- Asymmetric control relationships require global control flow
- The following global control mechanisms can be distinguished:
  - Independently controlled components
  - Dependent coprocessors
  - Incrementally controlled coprocessors
  - Partially dependent coprocessors

System Control Specialization

Dependent Coprocessors
**Incrementally Controlled Coprocessors**

- Embedded System Architectures
- Architecture Specialization Techniques

**Partially Dependent Coprocessors**

- Embedded System Architectures
- Architecture Specialization Techniques

**Application System Classes**

- Computation oriented systems
- Control dominated systems
- Data dominated systems
- Mixed systems

**Control Dominated Systems Architectures**

- Input MOCs are coupled FSM or Petri-Nets
- Co-design problems:
  - execution of concurrent FSMs/Petri-Nets reacting to asynchronous input events
  - FSM transition synchronization
  - event scheduling
  - correctness

**8051 - an 8bit Microcontroller Architecture**

- Embedded System Architectures
- Architectures for Control Dominated Systems

**Examples - Philips 80C552**

- Embedded System Architectures
- Architectures for Control Dominated Systems
Data Dominated Systems Architectures

- Data transport or processing is dominant
- Flow graph languages are used to describe such systems
- Opportunities for specialization:
  - Periodic execution often corresponds to a input data independent system data flow
  - Input data is mostly generated with a fixed period (the sample rate)

Examples - MMX

Examples - Hitachi SH-DSP

Examples - Philips Trimedia TM1000

Examples - TI TMS320C80

Customized Instruction-Sets

- Instruction-Set Architectures (ISAs) are the visible instructions of a processor
- Is there a strong motivation for customizing processors’ instruction-sets?
Barriers

- Existing binaries
- Toolchain costs and user familiarity
- Lost savings/higher chip cost due to lower volumes
- Hardware development costs
- The product development cycle for embedded products

Co-design Problems

- Instruction-set definition
- Instruction encoding for code compression
- Instruction encoding for memory optimization
- Global control and data flow optimization
- Communication channel selection
- Component interface synthesis
- Component selection/reuse

Conclusions